

# SimpleR - Handbook

(for SimpleR Version 0.1.0)

**Gunther Maier**  
**Wirtschaftsuniversität Wien**  
**Vienna, Austria**  
([gunther.maier@wu-wien.ac.at](mailto:gunther.maier@wu-wien.ac.at))

## 1. Why you may not need this handbook

SimpleR is intended to help Windows-users to start using R, the exceptionally powerful open source statistics package. So, SimpleR tries to do the following things:

- to provide an environment that is intuitively understandable for Windows-users (menus, forms, select boxes, etc.),
- to give you access to most commonly used statistical functions,
- to allow you to manage your data analysis projects.

There are a few things that SimpleR does not try to do:

- to give you access to the full scope of statistical functions available in R,
- to cover the full functionality of R,
- to run on other operating systems except Windows.

Because of the intention to provide Windows look and feel, SimpleR should be easy to use for an experienced Windows-user even without this handbook or the use of the program's (still very marginal) help system. You are encouraged to try things out. Hopefully, things will work as expected. If not, please report the problem to [gunther.maier@wu-wien.ac.at](mailto:gunther.maier@wu-wien.ac.at). Your feedback will help improve the functionality of SimpleR. The program is still in a very early stage of development. The good news, however, is that it builds upon two well tested components: R and R(D)COM. For more information and step-by-step instructions for installing SimpleR, see the README file (<http://www.sre.wu-wien.ac.at/SimpleR/readme.html>).

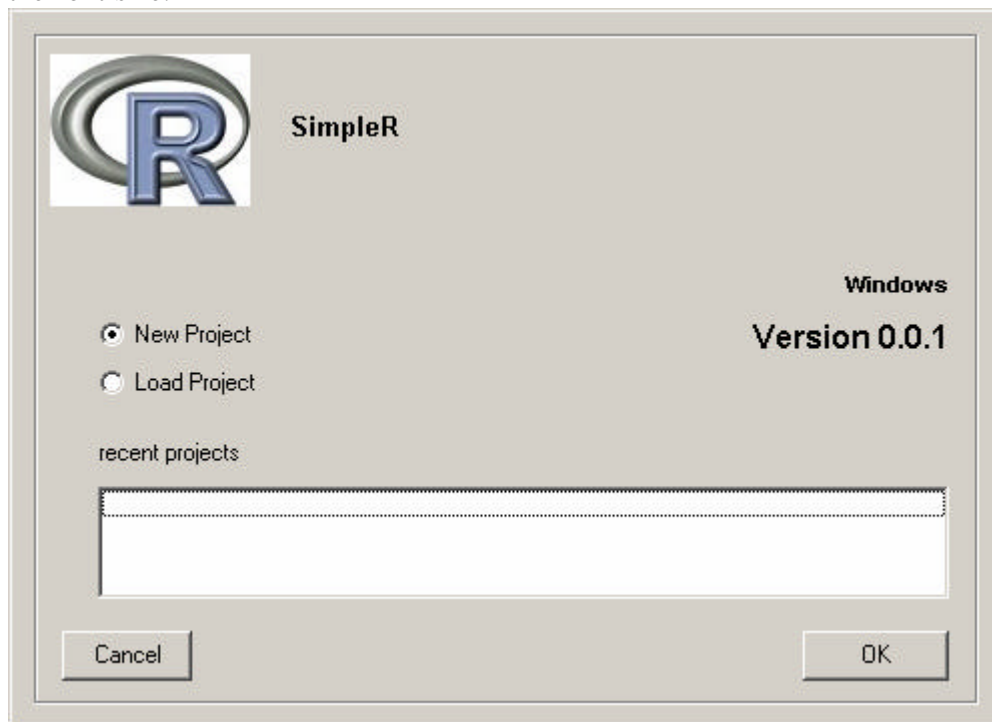
## 2. Starting the program

SimpleR is started in the usual way, by selecting it in the program menu or by double-clicking the SimpleR-icon on the desktop. When you start SimpleR, you do not get to its standard work environment immediately. First, you will see a startup window (fig. 1) that allows you to choose one of three options:

- you can start a new project – the default and the only meaningful option when you start SimpleR for the first time.
- you can open an existing project, i.e. open a project file that you have saved in an earlier session. This option will bring up the usual file-dialog for you to select the respective project file.

- you can open a recent project simply by selecting its project file from the list of recent projects. This list shows the ten most recently saved project files.

When you have made your selection, SimpleR brings up its standard work environment and either loads the selected project or starts with a new and empty project. When you start SimpleR for the first time, the program uses a small window size and an 8pt font. You can change the window size in the usual way. To change the font size, select Extras – Options from the menu. This will bring up the options form, where among other things you can select the font size.



**Figure 1**

SimpleR needs to do a number of tasks at this step. It needs to

- initialize some internal parameters,
- read the options and maybe the project file, and
- initialize R and the communication with R.

Particularly the last task takes time. So, be patient as it may take SimpleR a few seconds to complete all these tasks.

### **3. The SimpleR work environment**

Fig. 2 shows the standard work environment of SimpleR. It has four components:

1. the menu, which you will use for initiating most tasks in SimpleR;
2. the project window, which lists the data you use and the steps of your analysis;
3. the output window, which lists the results of your analyses;
4. the status area, which is used to give status information (currently this area is also used for some buttons, which are used for testing).

These components will be described in more detail below.

When the window of the work environment is enlarged, all the extra space is occupied by the project and the output window. You can change the division between the project and the output window by clicking and dragging the boundary between them to the desired location.

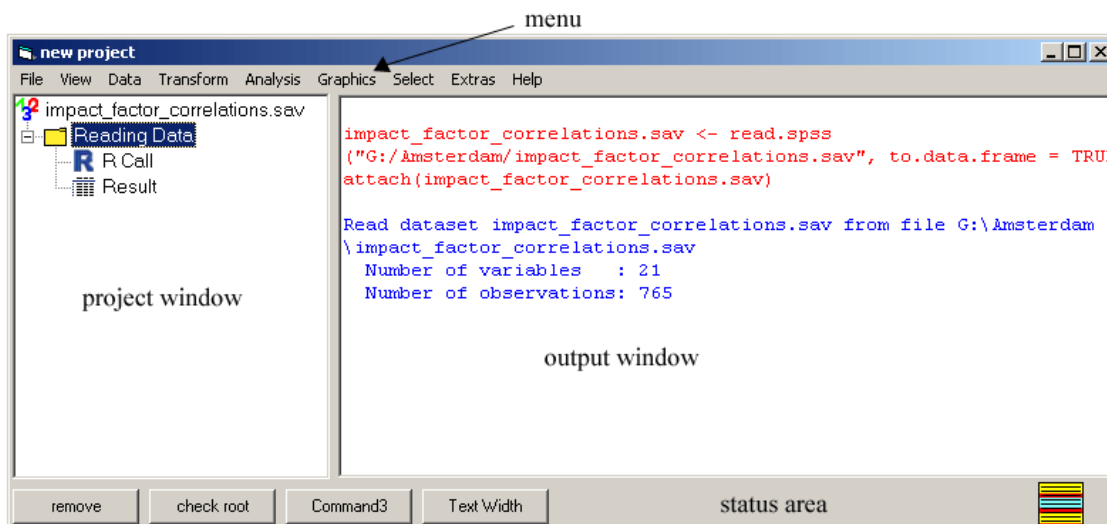


Figure 2

### 3.1. The SimpleR menu

The SimpleR menu is a fairly standard Windows menu. The only major exception is that you will not find any menu items for reading in data in the File – menu. These functions are concentrated in the Data – menu. The reason for this is that for SimpleR external data are just an entry in the respective project file. Consequently, you will find menu items for opening and saving projects in the File – menu.

The main entries in the menu are:

- **File:** contains entries for opening and saving project, for printing, for general information about the project and for exiting SimpleR.
- **View:** contains entries for viewing certain things, most importantly the dataset
- **Data:** contains entries for viewing, editing and reading data.
- **Transform:** contains entries for data transformations. These are computation of new data and the recoding of existing data . As of version 0.1.0 you can also extract a subset of observations to a new dataframe and merge two dataframes into one.
- **Analysis:** contains entries for all the statistical analyses implemented in SimpleR.
- **Graphics:** contains entries for the generation of various graphs.
- **Select:** contains entries for selecting a subset of the observations in a dataset or for subdividing the dataset.
- **Extras:** contains entries for extra tasks that do not fit into any other menu entry. Currently, these are the options and inserting a note.
- **Help:** contains entries to the help information.

### 3.2. The project window

The project window lists the components of a project in a tree-like structure. A project may have more than one sub-tree consisting of one or more nodes organized in hierarchical order. Every node except the root node has exactly one parent node, but may have any number of child nodes. The root of every sub-tree is a dataset. The first child is a data-header node (marked by a folder icon with a red “D”) that contains the information about the source of the data – where and how it was read or what data transformation generated the data, respectively. Commands that transform or recode data are called “datastep” commands. As long as no other child to the root node exists, such commands are accumulated in the data-header node. This means that the respective dataframe is altered rather than a new dataframe created. Therefore, the intermediate steps of data transformation are lost. However, the user can force the creation of a new dataframe in these commands.

At any time there is one node which is selected. When you click on one of the nodes in the project window, you make this node the selected node. Since the root node for every node is a dataset, by selecting a node you implicitly select a dataset as well. The root node of the selected node is the current dataset. All analytical steps in SimpleR are always applied to the current dataset.

When you select a node, the content of the output window is updated as well. The output window always shows the information corresponding to the selected node and all the nodes in the tree underneath it. So, when you select a dataset as the selected node, the output window shows the output of all the analytical steps you have applied to this dataset in the current project.

The tree underneath a node can be expanded or contracted by clicking on the plus or minus sign in the box to the left of a node, just like in Windows explorer. This allows you to fold away certain parts of your project and concentrate on others.

Nodes – and all the nodes in the tree underneath them – can be deleted from the project by pressing the Del-key. You will be asked to confirm this step. When the current node is a dataset, an extra warning will be issued.

SimpleR keeps track of the type of information that is stored in a node. This leads to different types of nodes. The most important types are

- Header: stores the headline for a specific analytical step.
- Data-Header: stores the headline for the datastep commands of a dataset
- RCall: stores the command that has been sent to R to perform this task
- Result: stores the text output resulting from an analytical step
- Graph: stores the link to the graphics output of an analytical step
- Note: stores the text of a user’s note (see below).

Normally, nodes are added to the project window automatically when you call some SimpleR procedures. However, you can add a note to the selected node by selecting Extras - Insert Note from the menu. A window will pop-up where you can enter any text you like. This text will be stored in a child node added to the current node.

### **3.3. The output window**

The output window lists the output stored for the currently selected node and for all the nodes in the tree underneath it. By default, the output of all types of nodes is shown in the output

window. However, you can restrict the output to certain types of nodes through the options. Select Extras – options from the menu to get a list of the node types. Deselect those that you do not want displayed in the output window. This setting is stored upon exit of SimpleR and loaded when you start the program again. So, when you are not interested in learning R-commands, for example, you can permanently filter out this information from display in the output window.

The output window usually list the text output stored in the nodes. The only exception is graph-nodes, which store the address of an image file that contains the respective graphics output. When a graph-nodes is selected in the project window, the graphics output is displayed in the output window. In all other cases (i.e., when the graph-node is in the tree underneath the selected node, only the address of the image file is shown.

SimpleR uses different colors for displaying the various types of information in the output window. R commands, i.e., information stored in RCall-nodes, are displayed in red. Results of the analyses, i.e., information stored in Result-nodes, are displayed in blue. All other information is displayed in black.

### 3.4. The status area

The status area is used for showing status information about SimpleR. Currently, the only status information shown in the status area is about selection of data. Through the menu item Select you can specify either a selection of specific observations from the dataset or the subsetting of the dataset according to the values of a variable. All subsequent analyses are then applied to only this subset of the data or to each group of data, respectively. A graphical indicator in the right hand side of the status area shows whether subsetting or subgrouping is in effect. The indicators are



when subsetting is in effect (only some lines are highlighted in the icon)



when subgrouping is in effect (all lines are highlighted, but in different colors)

When neither of these settings is in effect, the respective part of the status area is empty.

Currently, the status area is also used for various buttons that are used for testing purposes. These buttons will be removed in future versions of SimpleR.

## 4. Working with SimpleR

As has been mentioned in section 1, SimpleR intends to provide only the most important statistical functions. However, these functions are probably sufficient for a majority of users. If you need more specialized or sophisticated functions in R (and there are many available in basic R and in loadable packages), you can always switch from SimpleR to the regular R console and continue working there. In this case you will have to know the necessary R statements or at least know where to look them up. SimpleR, however, helps you in this transition in various ways:

- When you save a project, SimpleR always stores the current workspace in a file with the name of the project file and the extension “.RData”. This file contains all the data you have loaded and all the data transformations you have made in SimpleR. Once in the R console, you can load this workspace simple through the “load” command.
- SimpleR always tells you the commands it has generated for you and submitted to R. These commands are printed in red in the output window. By studying these commands, you can learn the syntax of R “on the job”. Also, you can copy-paste these commands from the SimpleR output window to the R console to re-run certain analytical steps in R.

SimpleR supports four basic steps that are used in almost every statistical project:

1. reading datasets
2. data transformation
3. data analysis
4. graphing of data.

These steps for working in SimpleR will be described in this section.

#### 4.1. Reading Datasets

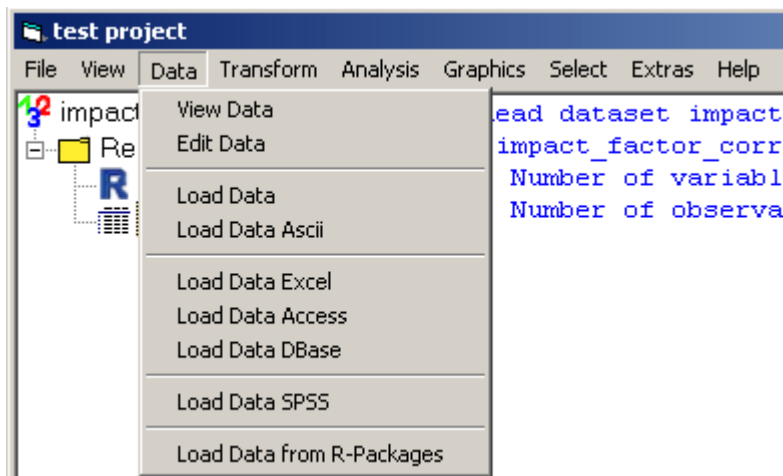


Figure 3

All the commands for reading datasets can be found under “Data” in the SimpleR menu. Fig. 3 shows the respective submenu. Currently, SimpleR supports the input of four types of data formats: ASCII, database formats, statistical packages formats and R-packages format.

Most of the functions behind these menu items are self explanatory and involve nothing more than changing to the right directory and clicking on the right file. This refers particularly to the functions “Load Data SPSS” and “Load Data from R-Package”. When loading database format files you typically have to make some more selections after selecting the right file. The reason for this is that a database file typically contains various tables, so that you have to pick the right table after selecting the right file.

The most general function for reading data is “Load Data Ascii”. It is also the most demanding one for the user to use. Therefore, this function is described in some detail here. Since the function tries to implement most of the options of R’s “read.table” command, the information about read.table in the R documentation provides additional information.

When you select “Load Data Ascii” from the “Data” menu in SimpleR, your first step is to select the appropriate file in the usual way. After this, the “Load Ascii Data”-form (fig. 4) appears that allows you to specify a large number of parameters. The form has two distinct areas. In the top part you can specify the parameters for reading the Ascii-file, in the bottom part the first few lines of the file are displayed for your information. There you can get a good idea of which parameters need to be set.

There are different data formats the form can handle by default. You can select the formats through the “Format” – select box. Upon startup the form shows the “Standard” format with the parameter settings as shown in fig. 4. Other formats that can be chosen from the “Format” – select box are “csv”, “delim”, “csv2”, “delim2”, and “fixed width”. These formats represent special parameter constellations for data input. The “fixed width” requires additional information that will be described below.

Figure 4

As compared to the “Standard” format, the settings of which are shown in fig. 4, the other formats imply the following changes:

- “**csv**”: Header: true, Separator Char: comma , Decimal Char: period , Quote Chars: double quote
- “**delim**”: Header: true, Separator Char: tabulator , Decimal Char: period , Quote Chars: double quote
- “**csv2**”: Header: true, Separator Char: semicolon , Decimal Char: comma , Quote Chars: double quote
- “**delim2**”: Header: true, Separator Char: tabulator , Decimal Char: comma , Quote Chars: double quote
- “**fixed width**”: Header: false, Separator Char: tabulator , Decimal Char: period , Quote Chars: single and double quote

These are predefined parameter constellations that you can change in any way that is necessary for correctly reading the data. SimpleR actually constructs a “read.table” command from the selected parameters and passes it along to R. For more information on the meaning of these parameters, see the document “R Data Import/Export” from the R documentation.

When you select “fixed width” as format, the lower part of the form changes to fig. 5. You see a broader gray area above the displayed data and a black vertical bar to the left of the display. You use these to define the fixed width format of the data you want to read. Grab the black bar on the left hand side of the display with the mouse and drag it to the locations of the boundaries between your variables. Make sure that the mouse is in the white data area when you release it. A small vertical bar will appear in the gray area above your data indicating the chosen boundary. An example can be seen in fig. 6.

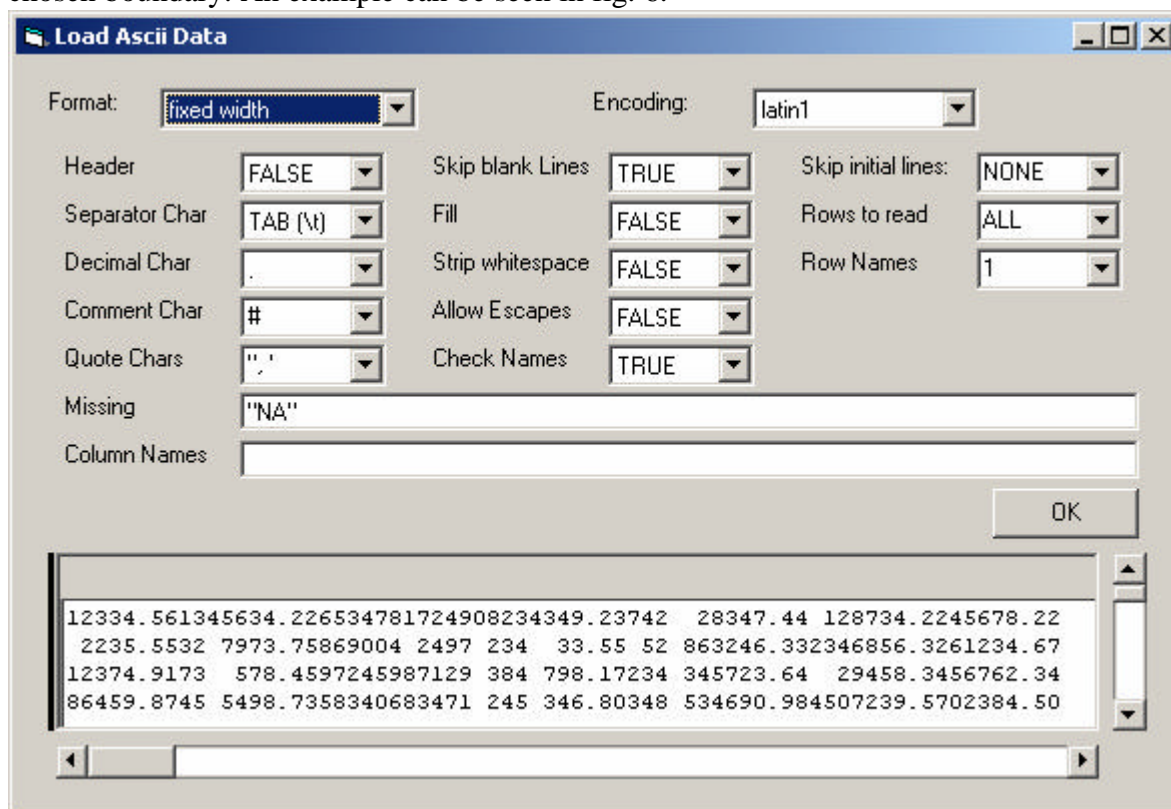
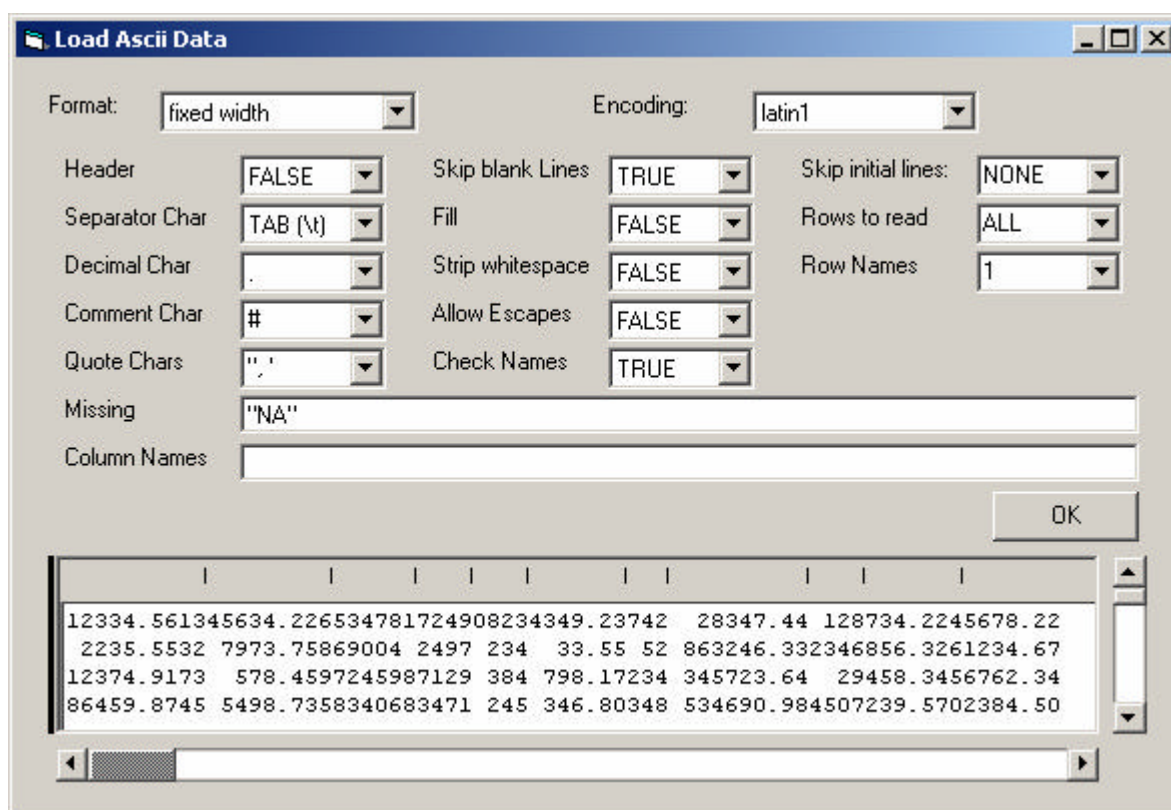


Figure 5



**Figure 6**

If you happen to misplace a boundary marker, you can remove it by dragging the bar to this location again.

When you click OK, SimpleR generates the appropriate R statement for reading the selected file and passes it along to R. In a small input form you will be asked to name the data. This name will be used by SimpleR to identify the data in the project window as well as by R to identify it in the workspace. When the data are read successfully, a new data node will appear in the project window.

## **4.2. Data Transformation**

A second important step in the analysis of data, once you have loaded a dataset, is to transform these data; to generate new variables based on existing ones, to change variables or to delete them. These are the three functions currently available from the Transform item in the SimpleR menu.

The Delete Variable function is very simple. When you move the mouse over this item in the menu, a submenu with the names of all the variables in the current dataset appears, where you select the variable to be deleted. To avoid unintentional deletion of variables, you will have to verify this step.

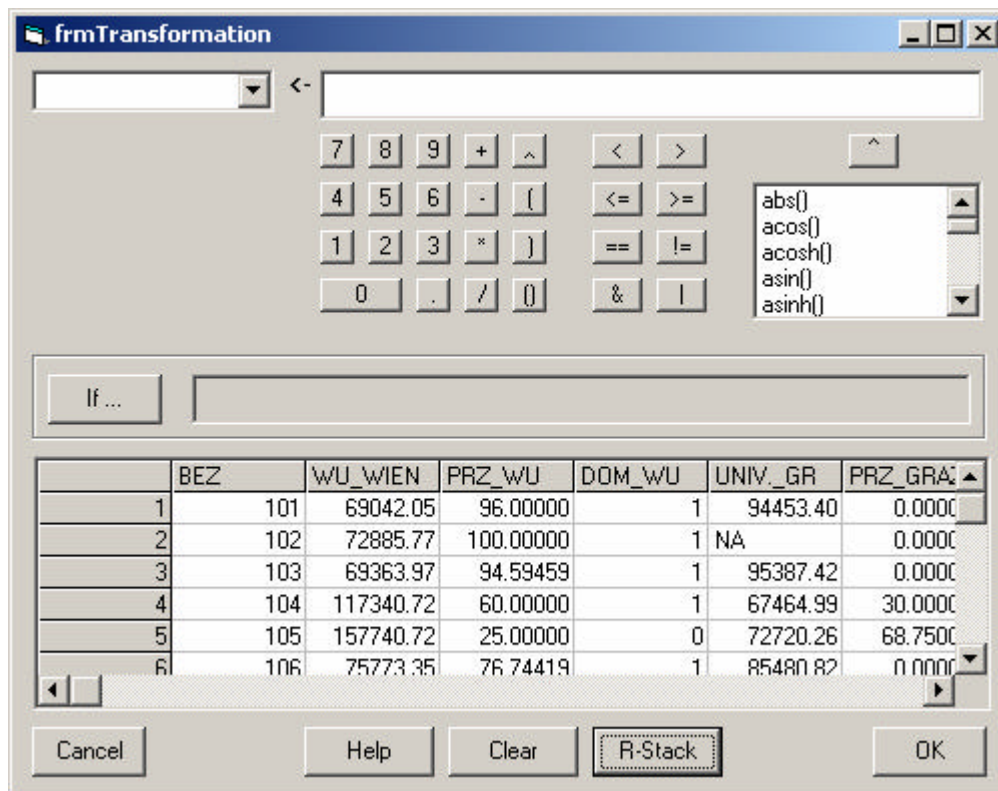


Figure 7

#### 4.2.1. Transformation

The other two functions, Transformation and Recode, may need some more detailed description. When you select Transformation from the menu, a form (fig. 7) opens up that lets you specify the data transformation you want.

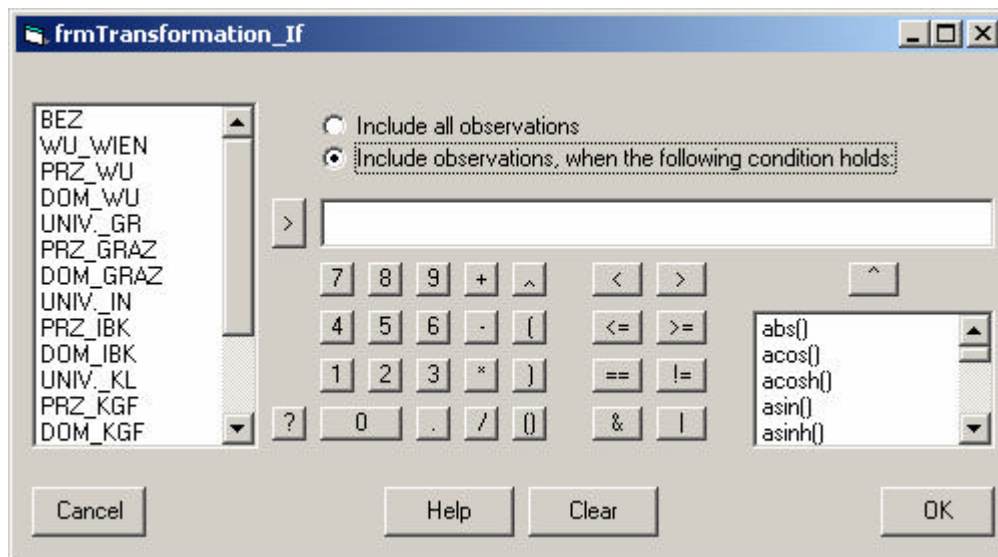
The form has five segments. From top to bottom these are:

1. The transformation function: In this section you can either directly specify the R statement for the intended transformation or see the current content of this function, if you choose to construct it via the functions supporting the generation of this function. This section has two elements separated by the R assignment symbol “<-“. This symbol means: Assign the result of the statement to the right of the symbol to the variable with the name to the left of the symbol. Therefore, in the combo box on the left you can either enter a new variable name, if you want to generate a new variable, or select the name of an existing variable from the list. In the input box to the right of the assignment symbol you need to supply a valid R function, either by typing it directly or by constructing it from the supporting functions.
2. The calculator: This section provides one of the functions supposed to help you constructing your transformation function. It gives a series of buttons that simulate a simple calculator. The first set of buttons from the left gives numbers, parentheses and basic arithmetic functions (addition, subtraction, multiplication, division and exponentiation). The next set of buttons supplies logical functions (less than, greater than, less equal, greater equal, equal, not equal, logical and, and logical or). To the very right of this section you find a list of functions, that you can select and copy to the transformation function by clicking the button “^” above the list.

3. The If-section: Sometimes, a data transformation should be applied only to the specific set of observations rather than to all. In this section you can specify a logical condition that has to be true for the data transformation to be applied. When you open the transformation form, the If-statement is empty and greyed out, meaning that the transformation applies to all observations. To specify a condition, click on the “If ...” button. We will describe the form for specifying this condition below.
4. The data grid: This section shows the first few observations in your dataset in the form of a data grid. This section helps you specifying your transformation function in two ways: On the one hand it shows your data so that you don't have to remember all the variable names, on the other hand, it enters the variable names into the transformation function for you. If you click a column in the data grid, the respective variable name is entered at the current cursor position into the right hand side of the transformation function.
5. The button section: In this section you find the buttons for this form. The “Clear” button empties the transformation function and allows you to start all over again. The “OK” and the “R-Stack” buttons are described below.

Sometimes, a certain data transformation cannot be done in one step, but requires two or more. For example, if you want to generate a variable that is 1 when another variable contains a valid entry and 0 in the case of a missing value. You may implement this transformation in two steps: first, you may generate the new variable as a vector of ones, second you may select the observations where the original variable has missing values through an If-condition, and change the new variable (for this subset of observations) to zeros. Because of this, the transformation function in SimpleR does not send a transformation function to R immediately, but collects it first in a stack (the R-Stack). It is sent to R only when the “OK” button is clicked on an empty transformation function. So, when you have entered your transformation function and clicked OK, the transformation form does not close, but your transformation function is moved to the R-Stack and the transformation function in the form is cleared. Now, you can either specify one or more additional transformation functions, or click the “OK” button again to finally execute your transformations. In this case, you will be asked to name the new data set and the respective data node will be created in the project window of SimpleR.

The “R-Stack” button allows you to investigate the functions you have already moved to the R-Stack. Once you have moved a transformation function to the R-Stack you cannot recall or delete it. The only thing you can do to avoid its execution is to click the “Cancel” button and start all over again.



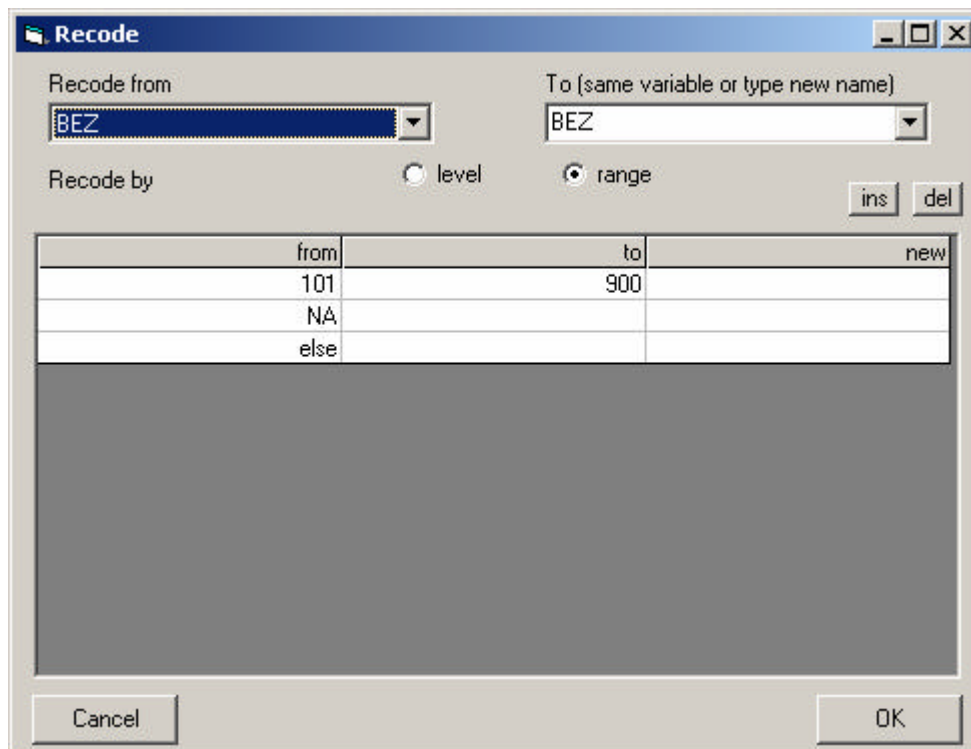
**Figure 8**

When you click the “If ...” button, a new form appears (fig. 8) that lets you specify the respective logical condition. Upon startup, the option “Include all observations” is active, which means that the condition – even where is one specified – has no effect. When you select the option “Include observations, when the following condition holds:”, the condition you enter in the input box underneath determines, to which observations your transformation function is applied. The functions for specifying the If condition are very similar to those of the Transformation form. The only differences are that the variable names are shown in a list and you can select a variable name from this list and enter it into the If condition by clicking the “>” button next to the list. The second difference is the “?” button. When you click it, a small form will open that shows the values and frequencies for the variable selected in the list. This function will be helpful when you want to select a specific observation.

When you click the “OK” button, the If condition you have specified in this form is copied into the respective input box of the Transformation form.

#### **4.2.2. Recode**

With the Recode function you can change the values of a variable from one value or range of values to a new value. When you select Recode from the menu, the Recode form (fig. 8) opens.



**Figure 9**

The form lets you specify the variable from which to recode as well as the one to which the new values should be written. While the source variable can only be selected from the list of existing variables, as destination you can either select an existing variable or specify a new one, simply by typing in the new variable name. By default, the name of the destination variable is set to that of the source variable.

The Recode function in SimpleR offers support for both categorical and continuous variables. Fig. 9 shows the settings for continuous variables. It can be explicitly chosen by selecting the option “range” in the “Recode by” line. Support for the recoding of categorical variables is chosen by selecting the option “level”. For variables that are of type factor in R, categorical recoding is selected by default, for all other types the default is continuous recoding.

As can be seen from fig. 9, the parameters for continuous recoding are supplied through a data grid. Upon startup, this data grid has two or three lines, depending on whether there are missing values in the respective variable or not. The first line gives the range of values in the variable, with the largest integer smaller or equal to the minimum in the “from” column and the smallest integer larger or equal to the maximum in the “to” column. If in the form displayed in fig. 9, you enter a value in the “new” column and click OK, all the values of BEZ between 101 and 900 will be recoded to this new value.

To specify more ranges of values, click the “ins” button. This will insert a new line in the data grid after the current line, which is the first line in this case. The lines with “NA”, if available, and “else” are always at the end of the data grid and cannot be deleted. In these lines, the content of the “from” and “to” columns cannot be edited. When a new line is added underneath the current line, the “to” value of the current line is shifted to the new line. This way, the minimum and maximum value are kept in the first and last line of the editable part of the data grid. However, this does not mean that you always have to cover the full range of values in your recoding statements, since you can click on any of these values and change it.

Through the line “NA”, if it exists, you can recode missing values in your variables. The line “else” determines the new values for those ranges that you have not specified explicitly. There are three possibilities:

1. You can leave the „new“ column of the „else“ line empty. In this case, values outside the specified ranges result in missing values.
2. You can specify a specific value. In this case, values outside the specified ranges are set to this value.
3. You can enter the character = in the „new“ column of the „else“ line. In this case, the original values remain unchanged, when they are outside the specified ranges.

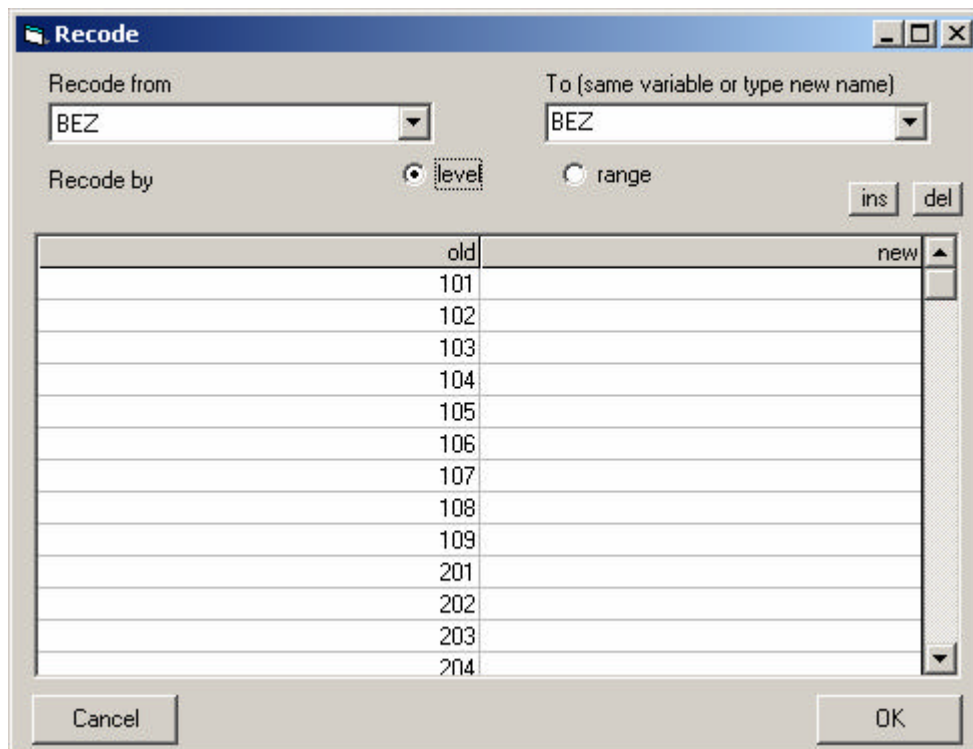


Figure 10

When you select the option „level“, the form changes to the one displayed in figure 10. Now the data grid has one line for every value of the selected variable – including the missing value indicator “NA”, if it applies, plus an “else” line. So, in this version you can specify a new value for every old value.

Irrespective of whether you use the level or range version of the “Recode” form, once you have specified all your recodings, you click the “OK” button. Before the respective commands are sent to R, you will be asked to enter a name for the new data set. When you enter a valid name, the commands are sent to R and a new data node with the name you have specified is created.

### 4.3. Data Analysis

Of course, data analysis is the core function of any such system as SimpleR. Currently, there are only three items in the Analysis menu: “Descriptive”, “Regression” and “Correlation”,

where “Descriptive” has its own submenu. More functions will be added to SimpleR, although no attempt will be made to cover the full functionality of R.

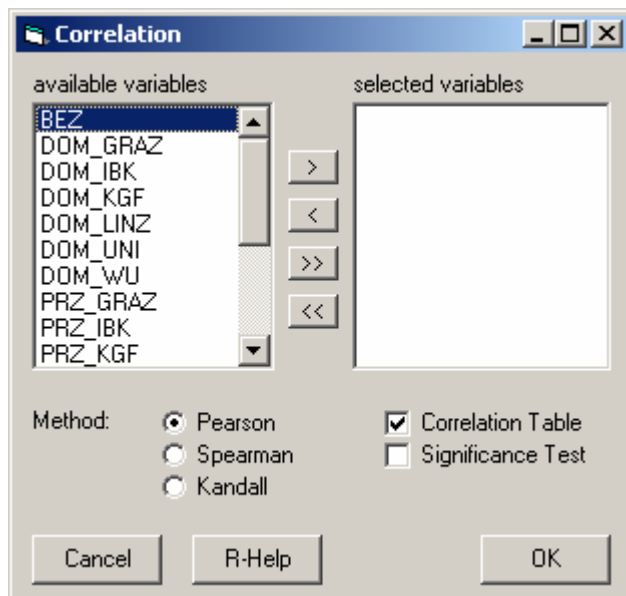


Figure 11

Data analysis with SimpleR is fairly straight forward. The steps involved are common to all functions: when you select any of the functions, a form will open. This form, like the one for correlation analysis which is shown in fig. 11, gives a list with all the available variables. When the statistical function requires a numeric variable, no factor variables will show up on this list. Then, depending on the function, there will be another list and/or one or more input boxes to which you select the variables. Since correlation analysis can handle any number of variables in any sequence, you select the variables to a list, in this case. Regression analysis, on the other hand, differentiates between the dependent variable and independent variables, the first one is selected to an input box, while for independent variables a list is used.

Also, you will find buttons for moving one variable to and from the selected area (“>”, “<”), with some functions also buttons for moving all variables (“>>”, “<<”). Single variables can also be selected and de-selected by double clicking their names. For some functions you will find additional options that you can set. In the case of correlation analysis you can select from three methods of calculating correlation, and you can specify whether tables with correlation coefficients and with significance tests should be generated.

#### 4.3.1. Data Analysis Functions

Since there is nothing special about data analysis in SimpleR, instead of a detailed description of every data analysis function, we will only give a quick overview of the available functions.

##### Descriptive

This function generates descriptive statistics for the selected variables. You can choose from eight descriptive measures: minimum, maximum, number, sum, mean, median, variance, and standard deviation. With the number also the number of valid observations and that of missing values are reported.

## Summary

This function gives similar information as “Descriptive”. For every numeric variable the function lists minimum, 1<sup>st</sup> quartile, median, mean, 3<sup>rd</sup> quartile, maximum and the number of missing values. For factor type variables the function gives names and frequency of the factors.

## Quantiles

This function reports the quantiles of numerical variables. In addition to selecting the variable(s) you can also either choose the number of groups you want to distinguish or the boundaries between the groups. In the first case, the boundaries are equally spaced, in the second you define the boundaries between the groups explicitly. The boundary values are specified as numbers between zero and one. For example, if you specify "0.2 0.6 0.8" as boundary values, the output will show four groups: first 20 percent, 20 – 60 percent, 60 to 80 percent, and last 20 percent.

## Crosstabs

With this function you can cross-tabulate one to three variables. In addition to the simple tabulation of the values, you can request margin sums, margin proportions, total proportions, a chi-square test of independence and the contributions of each table cell to the chi-square test.

## Regression

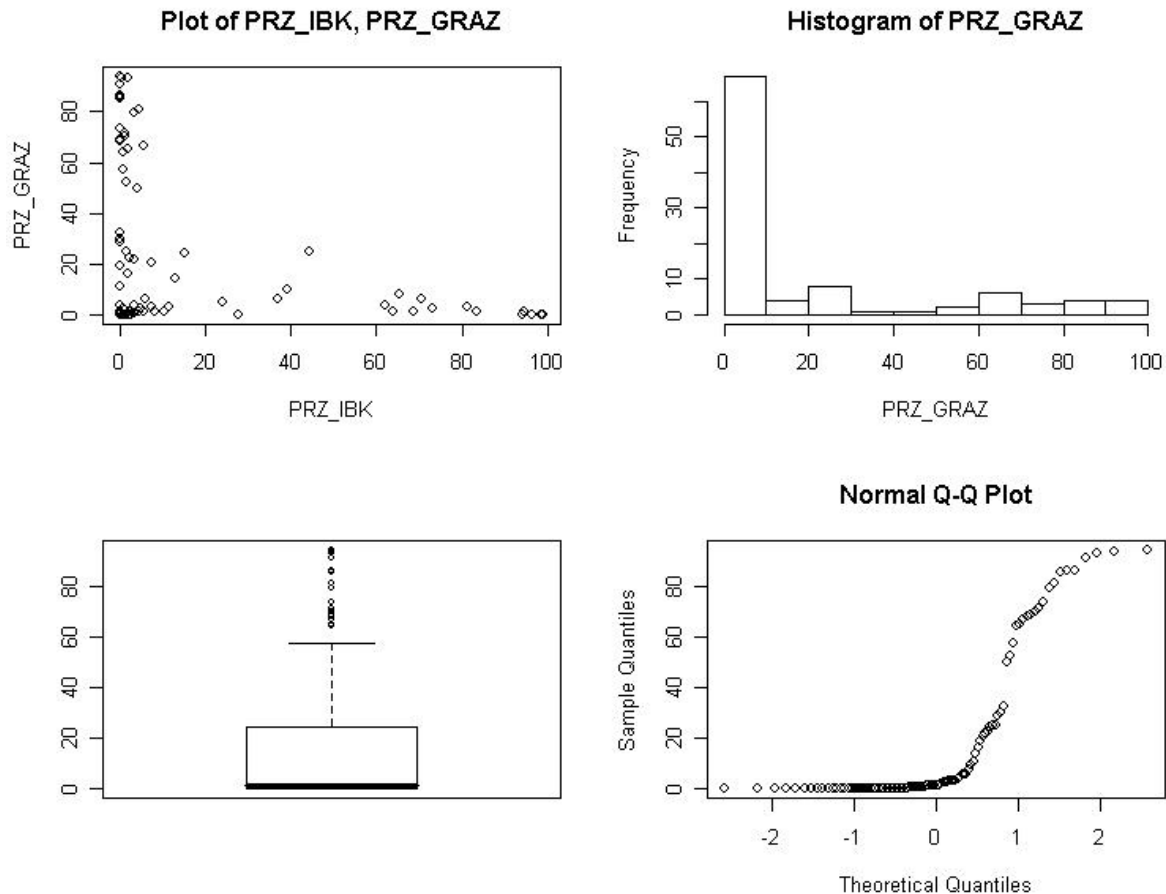
With this function you can do regression analysis and analysis of variance. You have to select a dependent variable and at least one independent variable. By default the regression summary is reported. In addition you can request lists of predicted values and residuals, analysis of variance, and R’s diagnostic plots. A model option allows you to suppress the intercept.

## Correlation

The Correlation function has already been described above. You can select from three methods of calculation – Pearson’s product moment correlation, Spearman’s rank correlation, and Kendall’s tau. By default only the correlation matrix is generated. In addition you can request a matrix of tests about the significance of the correlation coefficients.

### 4.4. Graphing of Data

An important step in the analysis of data is to display the information graphically. SimpleR implements the basic high-level graphics functions of R and allows you to generate various types graphs: various forms of dot and line plots, histograms, boxplots, and Q-Q-plots. Figure 12 shows the four types of high level graphics.

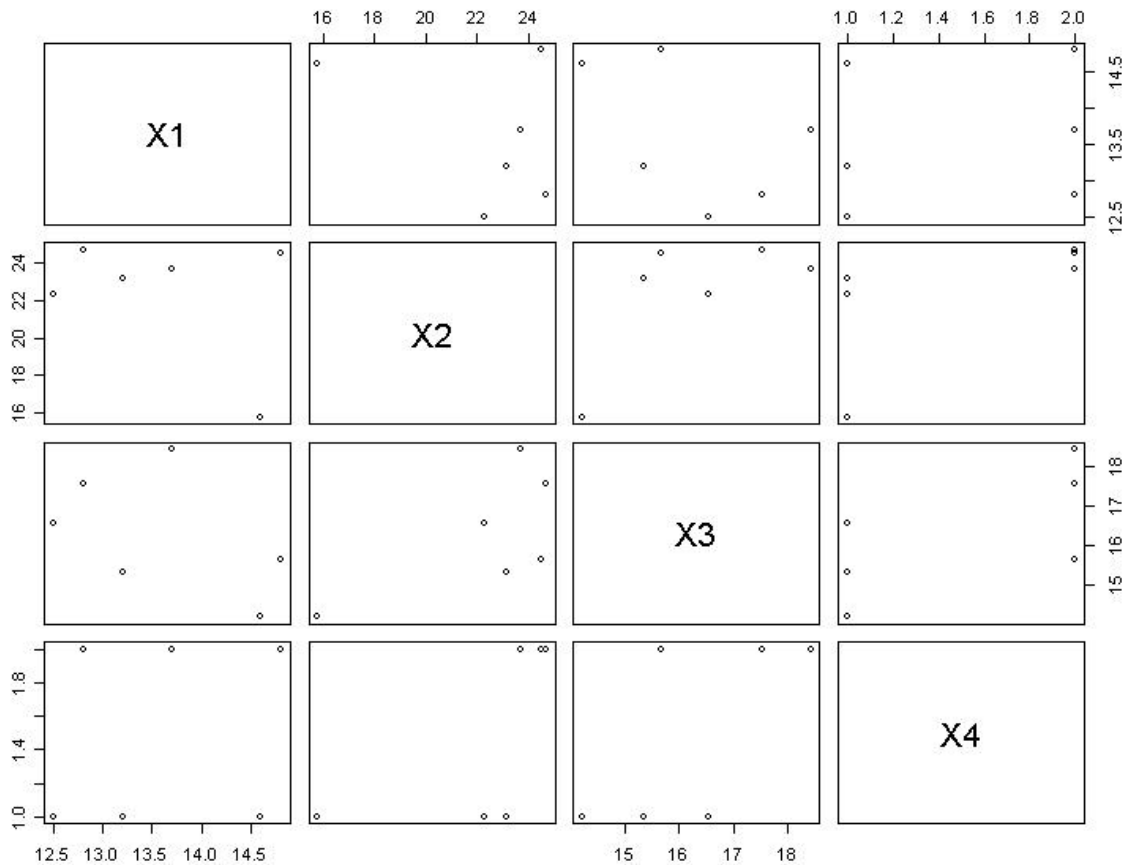


**Figure 12**

Some general aspects are important, when graphing information with SimpleR. All the functions we have dealt with so far have generated text output. Graphical information needs to be handled differently. Therefore, SimpleR has a special graphics mode – as compared to analysis mode – in which it operates when dealing with graphics. The program enters graphics mode automatically, when one of the functions requiring this mode is selected. When SimpleR is in graphics mode, you cannot use any of the analysis mode functions. Therefore, the respective menu items are disabled in graphics mode.

There is only one menu item that lets you make SimpleR exit graphics mode: “Close Graphics”. When the program is in analysis mode, the respective item in the Graphics menu is disabled. When the program is in graphics mode, you can select other high level graphics functions and low level graphics functions to add lines and text to the currently active graph. When you choose another high level graphics function, the currently active graph is usually replaced by the new one (we will mention an exception below). When in graphics mode, you can try out different types of graphs overwriting the old with the new one. When you exit graphics mode by selecting “Close Graphics” from the “Graph” menu, the current version of your graph is saved to a file on the harddisk of your computer. You cannot go back to this graph in SimpleR and change it, once you have exited graphics mode.

Most of the enabled items in the Graph menu switch SimpleR into graphics mode. The only exception is “Pairs”. This graphics function generates scatter plots for all pairs of variables in a data set (called “dataframe” in R). Although this function uses graphics mode in SimpleR, it exits the mode automatically upon completion. An example of this function is given in fig. 13.



**Figure 13**

The function “Par” is special in another way. It does not generate any graphics output itself, but allows you to subdivide the graphics window such that you can fit more than one graph onto it. When you select this function, a form opens that allows you to set the number of rows and the number of columns you want to partition the graphics window to. If you choose two rows and two columns, for example, as has been done when producing fig. 12, you will be able to fit four graphs into the graphics window. The next four calls of high level graphics functions will fill these four areas. This is the above mentioned exception to the rule that output of previous calls to high level graphics functions is overwritten by that of more recent ones. When you exit graphics mode, SimpleR switches back to the default setting of one row and one column.

With the two low level graphics function “abline” and “text”, you can add lines and description to any currently active graph. When you select “abline” from the Graph menu, a form opens where you can specify the location of the line you want to add. Fig. 14 shows this form. You can specify the coordinate of a horizontal or vertical line, or the parameters a and b, which will add a line “ $y = a + bx$ ” to the graph.

The function “text” works in a similar way. It opens the form shown in fig. 15, where you can either specify three variables for the X, and Y coordinates and for the labels to be added, or the X, and Y coordinates and the label text explicitly. Additionally, you can set the position of the text relative to the specified coordinates. By default, the center position is selected, meaning that the text will be positioned centered at the coordinates. Alternatively, you can position the text above, below, to the right, or to the left of the coordinates.

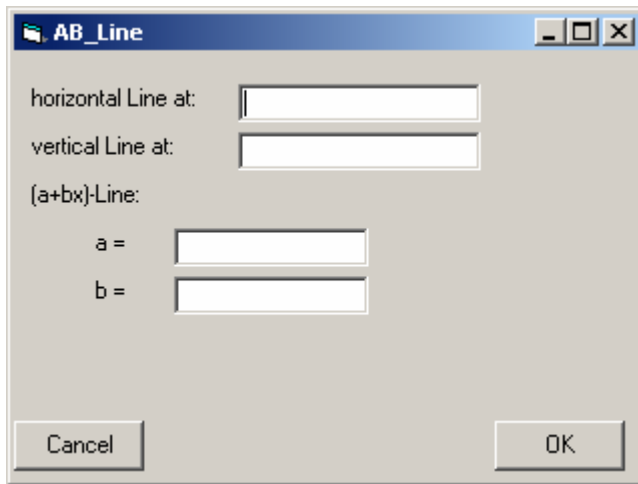


Figure 14

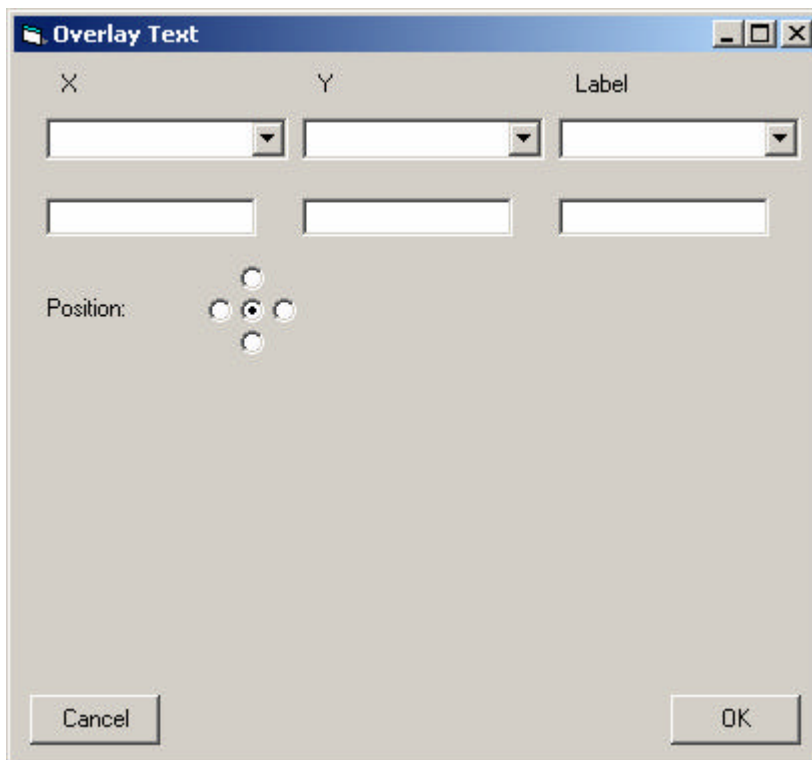
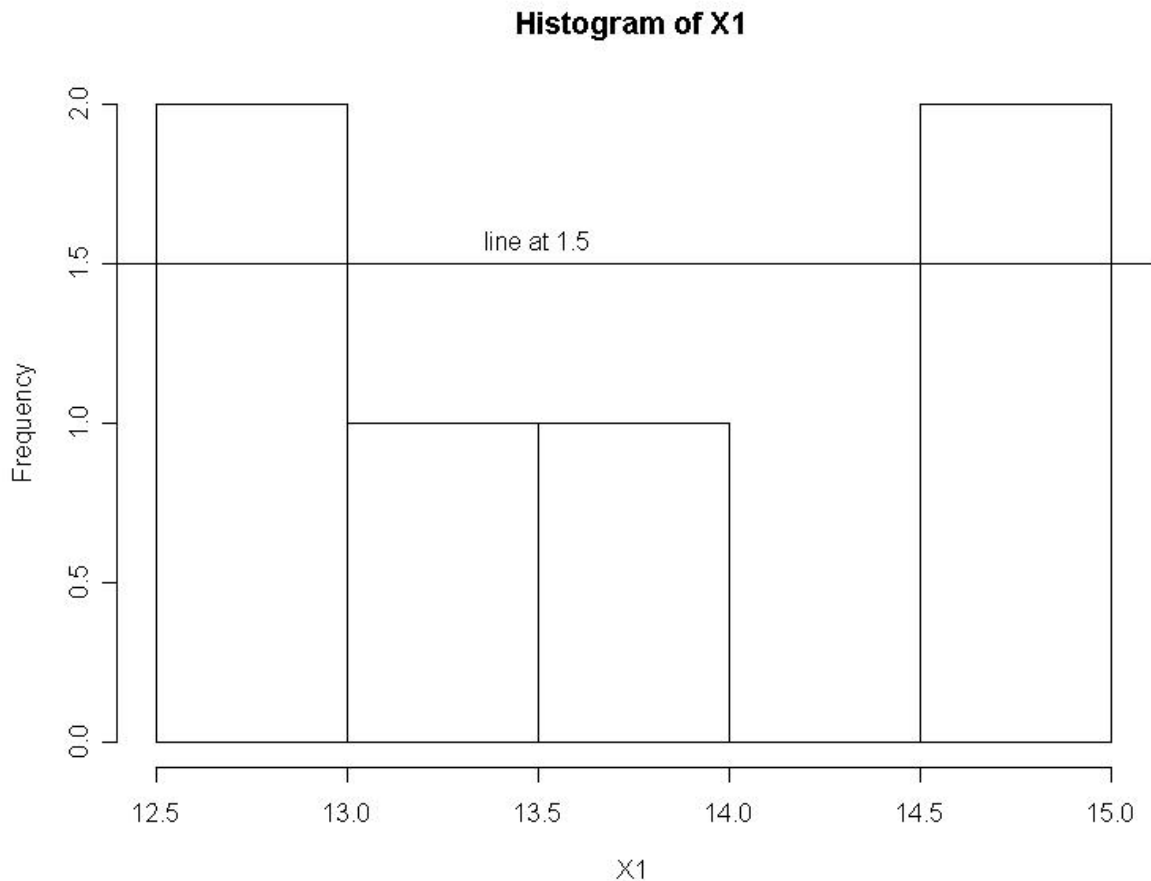


Figure 15


Figure 16 gives an example of a histogram with added line and text. We have added a horizontal line at  $y = 1.5$  and the text "line at 1.5" positioned above  $x = 13.5, y = 1.5$ .



**Figure 16**


#### 4.5. Subset and Group By

As mentioned above, SimpleR offers functions for subsetting and grouping of observations in your data set. Suppose you have information from a survey of population and want to analyze it only for the men in your data set. To do this, choose “Subset” from the “Select” menu. A form like the one for specifying If statements for data transformation (Fig. 8) opens, where you can specify your selection criteria. Select the gender variable, then the “==” button and

type the value that characterizes men. After clicking OK, the image  in the status area will indicate that all subsequent analysis and graphics commands will be applied only to a subset of the data. The subsetting criterion will also be indicated in the respective output.

To return to the full data set, select “Subset” again and disable the subsetting by choosing the option “Include all observations”.

If you want to do the same analyses separately for men and women, it is better to use the “Group By” function rather than repeat your analysis for two subsets. With the “Group By” function, SimpleR does this repetitive analysis for you. When you select “Group By” from the “Select” menu, a form opens up that shows the list of variables and lets you choose the one with the grouping criterion. In our example, this would be the gender variable. When you

click OK, the image  in the status area indicates that subsequent analysis and graphics

commands will be applied to groups of your data. The respective output indicates the variable used for grouping as well as the respective value.

To return to the full dataset, select “Group By” again and move the grouping variable back to the variable list.